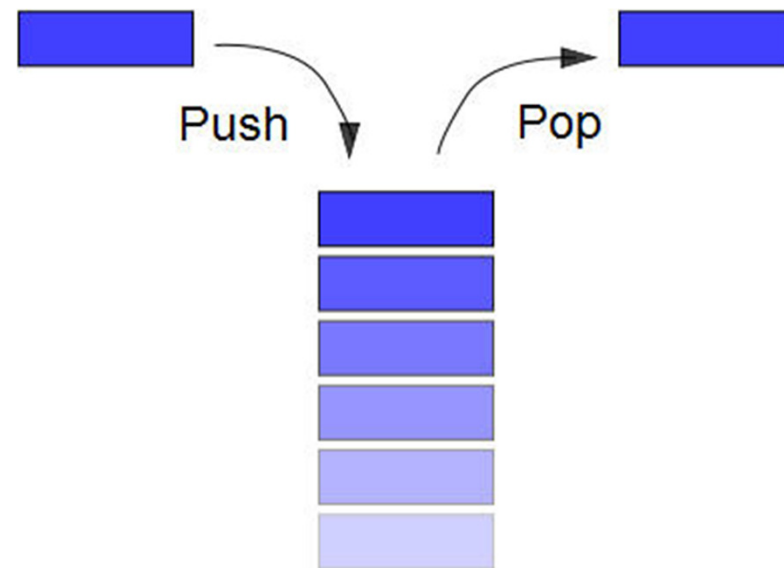


LAB#3

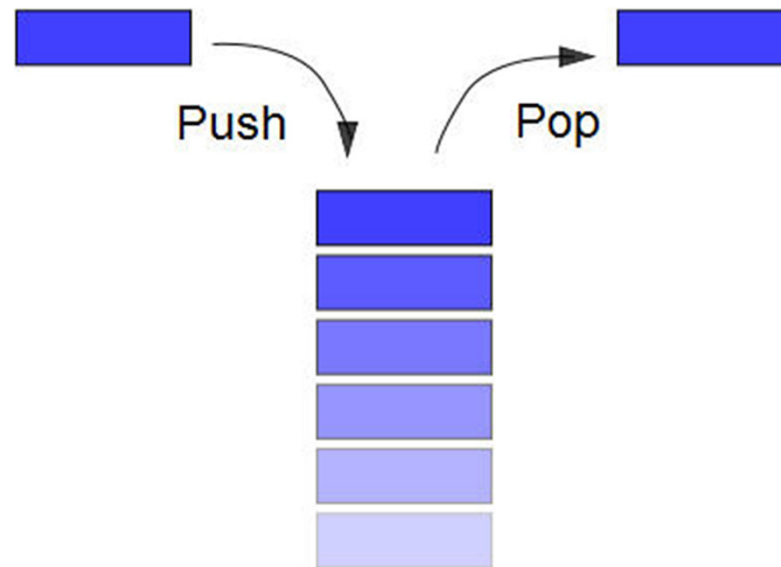
Stacks



Stacks

Stack:

- A *stack* is a last in, first out (**LIFO**) abstract data type and data structure.



Stacks

A Simple Stack Class:

- We use two classes: **Node** and **Stack**
 - Declare **IntNode** class for the nodes

```
class IntNode
{
public :
IntNode(int el, IntNode *ptr = 0) {info = el; next = ptr;}
int info;
IntNode *next;
};
```

Stacks

A Simple Stack Class:

- Declare **stack** which contains :

```
class stack {  
public:  
stack( ) {head = tail =0; }  
bool isempty( );  
void push(int);  
int pop( );  
int peek( );  
void Displaystack( );  
void clear( );  
private:  
IntNode *head, *tail; };
```

Stacks

A Simple Stack Class:

- Declare **stack** Class member function:

1- **bool isempty();**

```
bool stack::isempty( )
{
if (head==0 && tail ==0)
return 1;
else
return 0;
}
```

Stacks

A Simple Stack Class:

- Declare **stack** Class member function:
2- **void push(int);**

```
void stack::push(int data) {  
    IntNode *newnode;  
    newnode = new IntNode(data,0);  
    newnode->next = head;  
    head = newnode;  
    if (tail==0)  
        tail = head; }  
}
```

Stacks

A Simple Stack Class:

- Declare **stack** Class member function:

3- **int pop();**

```
int stack::pop( ) {  
    int val=0;  
    if(head!=0)  
    { val = head->info;  
      head = head->next; }  
    if(head==0)  
    tail=head;  
    return val;}  

```

Stacks

A Simple Stack Class:

- Declare **stack** Class member function:

4- **int peek();**

```
int stack::peek( )
{
if(!isempty( ))
return head->info;
}
```


Stacks

A Simple Stack Class:

- Declare **stack** Class member function:

5- **void Displaystack();**

```
void stack::Displaystack( )
{
    IntNode *current;
    current = head;
    while(current != tail->next)
    {
        cout << current->info << " " << current << "\n";
        current=current->next;
    }
    cout << "-----" << "\n";
}
```

Stacks

A Simple Stack Class:

- Declare **stack** Class member function:

6- **void clear();**

```
void stack::clear( )  
{  
head = tail =0;  
}
```

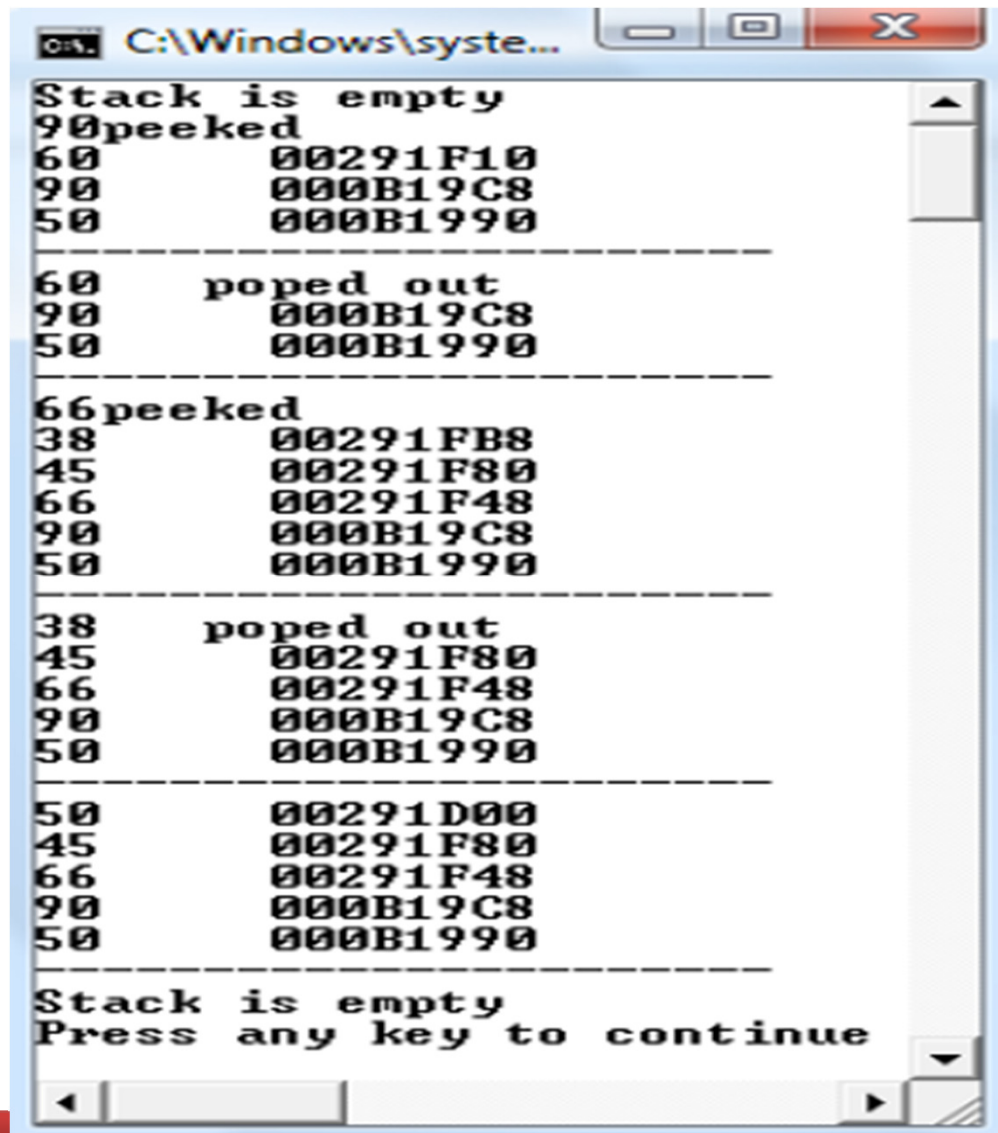
Stacks

A Simple Stack Class:

```
void main( ) {  
    stack mag;  
    if(mag.isempty( )) cout << "Stack is empty \n"; else cout << "Stack is not empty \n";  
    mag.push(50);  
    mag.push(90);  
    cout << mag.peek( ) << "peeked \n";  
    mag.push(60);  
    mag.Displaystack( );  
    cout << mag.pop( ) << " popped out \n";  
    mag.Displaystack( );  
    mag.push(66);  
    cout << mag.peek( ) << "peeked \n";  
    mag.push(45);  
    mag.push(38);  
    mag.Displaystack( );  
    cout << mag.pop( ) << " popped out \n";  
    mag.Displaystack( );  
    mag.push(50);  
    mag.Displaystack( );  
    mag.clear( );  
    if(mag.isempty( )) cout << "Stack is empty \n"; else cout << "Stack is not empty \n"; }
```

Stacks

A Simple Stack Class:



```
C:\Windows\system...
Stack is empty
90peeked
60      00291F10
90      000B19C8
50      000B1990
-----
60      popped out
90      000B19C8
50      000B1990
-----
66peeked
38      00291FB8
45      00291F80
66      00291F48
90      000B19C8
50      000B1990
-----
38      popped out
45      00291F80
66      00291F48
90      000B19C8
50      000B1990
-----
50      00291D00
45      00291F80
66      00291F48
90      000B19C8
50      000B1990
-----
Stack is empty
Press any key to continue
```